NOV/DEC 1989 ISSUE # 27

VZ 200/300

HUNTER VALLEY VZ JOURNAL

A VERY
MERRY
CHRISTMAS
AND HAPPY
NEW YEAR

PRODUCED BI-MONTHLY BY H.V.VZ.U.G.

EDITOR'S COMMENTS . . . . .

This publication was prepared on a STAR NX 1000 PRINTER using Dave Mitchell's Patch 3.3 with E & F Word Processor. HI & LO-RES screen dumps and listings were done using Larry Taylor's Printer Patch V1.4 available from VSOFTWAREZ while Patch3.3 is available from Hunter Valley VZ Users' Group.

H.V.VZ JOURNAL SUPPLEMENT -

NOTE - This supplement like the previous one is designed to be removed from the journal and placed with previous supplements and is numbered accordingly.

27/3 - CONDOLENCES . . . . .

The Hunter Valley VZ User's Group expresses its condolence and symphathy to the Newcastle earthquake victims as well as their families and friends. Their loss and sorrow has been extensive.

Our sincere symphathy also to yesterday's (17-1-1990) victims of the Perth earthquake as well. So far we don't know if any loss of life, injury or damage has occured there, never-the-less we as victims ourselves understand their suffering and extend our symphathy to them.

SINCERE THANKS . . . . .

My deepest appreciation and thanks for the many local, intrastate and interstate phone calls from members expressing their concern about my and my families well being since the earthquacke.

We were away in Melbourne during the Christmas-New Year period and so we were safe. Our house sustained only minor damage. A cracked brick front verandah and front and rear steps which all have to be replaced. I count myself among the lucky ones.

APOLOGIES . . . . .

As you'll notice by this issue the Journal is two months late and I hope you accept my sincere apologies. Unfortunately some injuries from the car accident are getting worse instead of better. Besides the aches and pains I'm finding it very difficult at times to concentrate and remember things which I find very annoying.

The next issue will be late as well and I wish to inform our subscribers that nobody will miss out on their subscription as they are charged per issue, not time. It is hoped that that the Journal will be able to catch up by mid year. Please bear with us.

FEBRUARY CLUB MEETING . . . . .

On February 2, 1990 Peter Hickman an intrastate member will demonstrate his version of serial interface for the VZ together with a modem. I believe Peter has made quite a breakthrough as the disk drive can be used for data storage now. Should have more details next issue. Most of the local members myself included are exited about Peter's impending demonstration at our club and it should be an informative and entertaining evening.

VZ MOUSE . . . . .

Gary Bulley, one of our cleverer members has a working prototype for the VZ and it will be demonstrated at the club and in due course a detailed project will appear in the Journal.

DISCLAIMER . . . . .

Every effort is made to insure the accuracy of information contained within be it general, technical, programming, etc. No responsibility can be accepted by Hunter Valley VZ Users' Group or the Author as a result of applying such information in practice.

Looking at these in binary it is -

11111111B	FFH	MASK 0
00111111B	3FH	MASK 1
00001111B	0FH	MASK 2
00000011B	03H	MASK 3
00000000B	00H	MASK 4

Clearly, a Logical Right Shift of FFH will provide the mask sequence.

For those astute readers, the question of a right to left sweep of the screens should now be occurring! Using the same two starting bytes (B1H and D8H) the sequence of bytes is -
B1H - B0H - B8H - 98H - D8H.
Similarly, the mask sequence is -
FFH - FCH - F0H - C0H - 00H.
A Logical Left Shift of FFH will achieve this. Check all of this out for yourself to confirm that I am not "spinning a yarn".

Well, that completes the algorithm to allow hi-res screens to pass across one another. What does the Assembler code look like?

Here goes!

```
;LEFT TO RIGHT SWEEP OF HI-RES SCREEN
SSCN EQU 7000H       ;START OF VRAM
SBUF EQU 0F000H      ;START OF REPLACEMENT SCREEN BUFFER
SZSC EQU 0800H       ;SIZE OF SCREEN
LLEN EQU 20H         ;NO. OF BYTES IN ONE SCREEN LINE
NLIN EQU 40H         ;NO. OF SCREEN LINES
;
     CALL SAVR       ;SAVE ALL REGISTERS TO STACK
     LD IX,SBUF      ;POINT TO INCOMING BYTE
     LD IY,SSCN      ;POINT TO REPLACED BYTE
     LD B,LLEN       ;SET COLUMN COUNTER
NCL7 PUSH BC         ;SAVE COLUMN COUNTER
     LD H,0FFH       ;PIXEL MASK TEMPLATE
     LD B,4          ;SET PIXEL COUNTER
NPX7 PUSH BC         ;SAVE PIXEL COUNTER
     SRL H           ;SHIFT MASK FOR R.H. PIXEL PRESERVATION
     SRL H
     LD A,H          ;PUT MASK INTO ACC.
     CPL             ;NOT MASK
     LD L,A          ;NOT MASK IN L-REG FOR L.H. PIXEL PRESERVATION
     LD B,NLIN       ;SET LINE COUNTER
NLN7 LD A,(IX+0)     ;PUT INCOMING BYTE INTO ACC.
     AND L           ;MASK OUT R.H. PIXELS
     LD D,A          ;SAVE L.H. PIXELS
     LD A,(IY+0)     ;PUT REPLACED BYTE INTO ACC.
     AND H           ;MASK OUT L.H. PIXELS
     OR D            ;LOGICAL ADD R.H. AND L.H. PIXELS
     LD (IY+0),A     ;UPDATE SCREEN
     LD DE,LLEN      ;INCREMENT BY ONE LINE
     ADD IX,DE       ;POINT TO NEXT LINE OF INCOMING
     ADD IY,DE       ;POINT TO NEXT LINE OF REPLACED
     DJNZ NLN7       ;SEE IF LINES FINISHED?
     CALL DLAY       ;DO A PAUSE AT END OF COLUMN
     LD DE,0-SZSC    ;DECREMENT TO RETURN TO TOP OF CURRENT COLUMN
     ADD IX,DE       ;POINT TO TOP OF CURRENT COLUMN
```

```
        POP BC          ;RECOVER PIXEL COUNTER
        DJNZ NPX7       ;SEE IF ALL PIXELS DONE?
        INC IX          ;POINT TO NEXT COLUMN
        INC IY          ;POINT TO NEXT COLUMN
        POP BC          ;RECOVER COLUMN COUNTER
        DJNZ NCL7       ;SEE IF COLUMNS FINISHED?
        CALL RESR       ;RECOVER REGISTERS
        RET             ;FINISH
```

WELL THERE IT IS! I AM NOT GOING TO PROVIDE THE ASSEMBLER CODE FOR THE RIGHT TO LEFT SWEEP AS IT IS A SIMPLE VARIATION ON THE CODE GIVEN.

INCIDENTLY, NINE TYPES OF HI-RES SCREEN REPLACEMENTS ARE DETAILED IN AN ARTICLE BEING PRESENTED IN JOHN D'ALTON'S LE'VZ. LOOK OUT FOR THEM. ALONG WITH MY ARTICLE ON SCREEN MOVE SUBROUTINES IN HVVZUG NEWSLETTER, THERE IS NOW AN INTERESTING SET OF GRAPHICS AND ANIMATION HANDLING PRIMITIVES AVAILABLE FOR VZ USERS.

MY NEXT CONTRIBUTION IN THIS NEWSLETTER WILL CONSIST OF AN INTERRUPT-DRIVEN TECHNIQUE FOR REMOVING THAT ANNOYING HASH OR FLICKER FROM THE VZ SCREEN. (EVER PLAYED DSE'S DAWN PATROL?) THIS WILL COMPLEMENT MY SET OF CONTRIBUTIONS ON THE HI-RES GRAPHICS THEME.

IN THE MEAN TIME HAVE FUN WITH LOGIC OPERATORS AND ASSEMBLER.

LIVEN-UP ANIMATION & GRAPHICS BY BOB KITCH

7 EURELLA STREET KENMORE QLD 4069

WANT TO HAVE A UTILITY THAT DISPLAYS HIGH SPEED ANIMATION AND GRAPHICS WITHOUT FLICKER? THEN THIS IS THE PROJECT FOR YOU! READ ON!

THIS ARTICLE IS THE MOST COMPREHENSIVE AND INTEGRATED SOFTWARE PROJECT FOR THE VZ EVER ATTEMPTED IN A USERS GROUP MAGAZINE. THE PROJECT CONSISTS OF FIVE PROGRAMS - THREE WRITTEN IN BASIC (SET-UP, LOAD-UP AND THROW-UP) AND TWO WRITTEN IN ASSEMBLER (START-UP AND MOVE-UP) - RESULTING IN A PACKAGE THAT WILL "LIVEN-UP" THE USE OF THE VZ. THE GROUP OF PROGRAMS PROVIDE A UTILITY IN, WHAT I TERM, FAST BASIC. ASSEMBLER IS USED TO SPEED UP FUNCTIONS THAT ARE TOO SLOW IN BASIC. THIS MOST OFTEN OCCURS IN TWO INSTANCES - THE FIRST IS WHEN HIGH-SPEED PROCESSING IS REQUIRED, AND THE SECOND IS WHEN PRECISE CONTROL OF A PERIPHERAL IS REQUIRED. EACH ASSEMBLER PROGRAM FULFILS ONE OF THESE SHORTCOMINGS OF BASIC. (MOVE-UP AND START-UP RESPECTIVELY). NINE ASSEMBLER "PRIMITIVES" FOR SCREEN MOVES, GRAPHICS OR ANIMATION ARE PROVIDED IN MOVE-UP. THEY PROVIDE AN INTERMEDIATE INTRODUCTION TO ASSEMBLY LANGUAGE PROGRAMMING.

FOR THOSE WHO CAN'T WAIT FOR THE ENTIRE GROUP OF LISTINGS TO BE PUBLISHED OVER THE COMING MONTHS, (OR CAN'T BE BOTHERED KEYING THEM IN) I AM WILLING TO PROVIDE THE ENTIRE PROJECT ON DISK FOR $20. THIS INCLUDES ALL SOURCE AND OBJECT CODE PLUS A SET OF PICTURE FILES.

BACKGROUND TO PROJECT.

FOR SOME TIME NOW I HAVE BEEN CONTEMPLATING SOME REAL-TIME HI-RES GRAPHICS DISPLAYS ON THE VZ TO LIVEN-UP INTEREST IN THE COMPUTER. ALSO, IN A PREVIOUS ARTICLE, (SEE LE'VZ #24, PG. 3 & 4) I HAVE PROVIDED DETAILS OF HI-SPEED SCREEN MOVE SUBROUTINES. FURTHERMORE, THE ADDITIONAL MEMORY CAPACITY OF THE 64K MEMORY EXPANSION, THAT SOME USERS HAVE, HAS NOT BEEN GREATLY UTILIZED BY PROGRAMMERS. THIS ARTICLE, ACCOMPANIED BY THE SUITE OF PROGRAMS, PROVIDES SOME INSIGHT INTO THE THREE ASPECTS JUST DETAILED.

THE PROGRAMS ALLOW A PICTURE SHOW TYPE OF DISPLAY TO BE IMPLEMENTED WITHOUT CONSTANT UP-LOADING FROM DISK. THE PROGRAMS ARE WRITTEN FOR VZ'S THAT HAVE 64K MEMORY EXPANSION PACKS AND ARE DISK-BASED - ALTHOUGH THE PROGRAMS COULD BE EASILY ADAPTED TO VZ'S HAVING 16K EXPANSIONS AND TAPE UNITS.

FOR THOSE INTERESTED, THE SOFTWARE PACKAGES USED TO DEVELOP THIS UTILITY WERE AS FOLLOWS -

-STANDARD BASIC AND DOS ROMS.
-THE EDASM USED WAS DSE'S MODEL PATCHED WITH DISKOPS4.
-QUICKWRITE TEXT ED ALLOWS DISKOPS SOURCE FILES TO BE HANDLED AS W-FILES.
-THE MONITOR USED WAS FROM DISK DOCTOR AND WAS USED FOR PATCHING AND BREAKPOINTING.
-PICTURE FILES WERE BUILT USING ART GALLERY.

DESIGN CONCEPT.

A HI-RES ,SCREEN ON THE VZ OCCUPIES 2048 BYTES (2K) OF MEMORY. THE 64K MEMORY EXPANSION MODULE CONTAINS FOUR BY 16K BANKS OF RAM MEMORY. WHEN PROGRAMMING IN BASIC, ONLY MEMORY BANKS 0 AND 1 ARE EASILY ACCESSED. ASSEMBLY LANGUAGE PROGRAMMERS CAN ACCESS BANKS 2 AND 3, WHICH, IN FACT, OVERLAP IN ADDRESS SPACE WITH BANK 1. A FORM OF "MEMORY BANK SWITCHING" IS USED TO OVERCOME THIS OVERLAY PROBLEM.

WHEN PROGRAMMING IN BASIC, THE PROGRAM STATEMENT TABLE AND VARIABLE LIST TABLE BOTH RESIDE IN LO-MEM (GENERALLY IN BANK 0 ALTHOUGH LARGE PROGRAMS EXTEND INTO BANK 1). THE SYSTEMS AREA, CONSISTING OF THE VIDEO DISPLAY AREA AND COMMUNICATIONS AREA, ALSO RESIDE IN BANK 0. THE DOS VECTOR, STRING AREA AND STACK ARE ALL ORGANISED, ON BOOT-UP, INTO HI-MEM LOCATED AT THE TOP OF BANK 1. THE INTERVENING AREA OF RAM IS FREE SPACE, AND AS PROGRAM DEVELOPEMENT AND EXECUTION PROCEED, THE VARIOUS TABLES (WITH SOME EXCEPTIONS) DYNAMICALLY BUILD INTO THIS FREE SPACE. SWITCHING BANKS 1/2/3, WHEN IN BASIC, NORMALLY RESULTS IN A CRASH, AS THE HI-MEM TABLES BECOME "DISMEMBERED" FROM THE LO-MEM TABLES.

IT IS POSSIBLE TO REORGANIZE AND RESTRICT THE "BASIC WORK AREA" ENTIRELY INTO BANK 0 - PROVIDED THAT THE PROGRAMS AND THEIR ASSOCIATED DYNAMIC AND STATIC TABLES ARE KEPT SHORT. MEMORY BANKS 1 TO 3 ARE THEN "RESERVED" FOR OTHER USE SUCH AS VIDEO SCREEN BUFFER AREAS. THE BASIC WORK AREA EXTENDS FROM 7AE9H TO BFFFH. THE PROBLEM OF SWITCHING BANKS WHILST RUNNING BASIC PROGRAMS BECOMES TRIVIAL AND HI-SPEED MOVES FROM HI-MEM TO THE VIDEO RAM AREA BECOME POSSIBLE BY SHORT MACHINE LANGUAGE CALLS. IT IS ALSO NECESSARY FOR THE BASIC PROGRAM TO BE ABLE TO PASS VARIABLES TO THE M/L ROUTINE - BUT THIS IS ALSO EASY TO ACHIEVE. (THE LOAD MAP FOR THE PROJECT IS PROVIDED IN THE REMARKS ON THE END OF SETUP)

THE TOP OF BANK 0 IS BFFFH. BANKS 1 TO 3 OCCUPY C000H TO FFFFH. IT IS POSSIBLE TO BUFFER EIGHT 2K HI-RES SCREENS IN EACH MEMORY BANK, MAKING A TOTAL OF 24 SCREENS THAT CAN BE STORED IN THE 64K MEMORY PACK. THE POWERFUL Z80 BLOCK MOVE ROUTINE CAN MOVE 2K OF MEMORY AROUND IN 12.16 MSEC. (THAT'S FAST!) WITH THE Z80 RUNNING AT 3.58 MHZ. THE SCREEN DISPLAY IS UPDATED EVERY 20 MSEC. ALTERNATIVELY, SOME 96 LO-RES SCREENS (512 BYTES) COULD BE STORED INTERNALLY. OBVIOUSLY LOADING THE 48K OF DATA INTO THE MEMORY IS BEST ACHIEVED ON A DISK-BASED SYSTEM. ROUTINES TO HANDLE THE LOADING OF DATA AND THE SHIFTING OF DATA BETWEEN HI- AND LO- MEM (BOTH DIRECTIONS) ARE REQUIRED.

THE ABILITY TO MOVE DATA AROUND IN RAM AT THIS SPEED ALLOWS ANIMATED GRAPHICS TO BECOME A POSSIBILITY. THE TECHNIQUE IMPLIED IS NOT REAL-TIME STUFF HOWEVER. THE METHOD REALLY USES "PRE-FORMATTED" SCREENS OR "PAGES" THAT ARE PAGED INTO THE VIDEO DISPLAY AREA.

27/7 LIVEN-UP ANIMATION & GRAPHICS CONT.

I AM SURE EVERYONE WHO HAS USED THE VZ IN HI-RES MODE WILL HAVE NOTICED THE ANNOYING "FLICKER" ON THE SCREEN WHENEVER AN IMAGE IS UPDATED. THE CAUSE OF THIS EFFECT IS THE SUBJECT OF A FURTHER ARTICLE. IT IS SUFFICIENT TO SAY HERE, THAT THE FLICKER OCCURS WHENEVER THERE IS A CONFLICT BETWEEN THE TIMING OF A Z80 WRITE OPERATION AND A 6847 VDG READ OPERATION. BY INTERCEPTING THE INTERRUPT SIGNAL (PREVIOUSLY EXPLAINED IN LE'VZ #23, PG. 10 & 12 AND LE'VZ #24, PG. 8 & 9) AND USING A "SCREEN BUFFER AREA", IT IS POSSIBLE TO OVERCOME THE "SCREEN HASH" PROBLEM.

PROGRAM DESCRIPTION.

IT WAS DECIDED THAT THREE BASIC SUBPROGRAMS COULD ACHIEVE THE DESIRED EFFECT AND KEEP PROGRAM LENGTH SHORT.

THE FIRST BASIC PROGRAM IS CALLED SETUP AND DISPLAYS A SERIES OF INTRODUCTORY SCREENS TO THE USER. SETUP THEN CALLS THE FIRST OF THE ASSEMBLER ROUTINES CALLED STARTUP.

STARTUP CARRIES OUT A NUMBER OF FUNCTIONS THAT ARE MORE EASILY ACHIEVED FROM A LOW LEVEL LANGUAGE. AS THIS ROUTINE IS ONLY NEEDED TEMPORARILY, IT IS LOADED INTO PART OF THE HI-RES VIDEO RAM AREA. FIRST THE TOM IS LOWERED TO BFFFH AND ALL BASIC POINTERS ARE RESET ACCORDINGLY. NEXT, THE 310 BYTE DOS VECTOR IS PLACED BELOW THE NEW TOM BY JUMPING TO 4004H IN THE DOS ROM. THIS INVOKES A REBOOT OF THE SYSTEM AND THE READY MESSAGE WILL APPEAR. CONTROL IS PASSED BACK TO THE BASIC ROM AND THE DOS VECTOR AT 79ACH CALLED. TO AUTORUN THE NEXT BASIC PROGRAM CALLED LOADUP REQUIRES A SMART BIT OF PROGRAMMING AND KNOWLEDGE OF THE DOS ROM. THE SOURCE CODE FOR STARTUP DETAILS ALL. LOADUP IS BOOTED BY STARTUP.

THE SECOND BASIC PROGRAM CALLED LOADUP IS RUN NEXT. A NUMBER OF CHECKS ON THE CONFIGURATION OF THE VZ SYSTEM ARE CARRIED OUT TO CONFIRM THAT ALL IS O.K. THE BANK SWITCHING OF THE 64K PACK IS EXERCISED AND VERIFIED AS PRESENT AND OPERATIONAL. NEXT, THE FIRST 4 IDENTIFICATION BYTES (AAH, 55H, E7H, 18H) OF THE DOS ROM ARE VERIFIED TO CONFIRM THAT A DISK-BASED SYSTEM IS IN PLACE. THE BOOT-UP OF THE SYSTEM, CARRIED OUT BY THE INITIAL PROGRAM LOADER, SEEKS THESE FOUR BYTES AT 4000H, 6000H AND 8000H TO LEARN THE CONFIGURATION OF THE SYSTEM BEING BOOTED. ANY ROM CARTRIDGE INSERTED AT THESE LOCATIONS NEEDS THESE IDENTIFICATION BYTES TO BE RECOGNIZED. AFTER THIS, THE TOP-OF-MEMORY LOCATION UNDER THE DOS VECTOR IS CHECKED SO THAT NO CONFLICT ACROSS THE BANK 0 TO BANK 1/2/3 ADDRESS CAN OCCUR.

LOADUP THEN FURTHER LOWERS THE TOP-OF-MEMORY TO AFFFH (THE DOS VECTOR IS NOT MOVED). THIS CREATES AN AREA OF "PROTECTED MEMORY". THE SECOND OF THE ASSEMBLER ROUTINES CALLED MOVEUP IS LOADED IN TO THIS AREA OF MEMORY. A SMALL M/L ROUTINE IS ALSO LOADED INTO PROTECTED MEMORY. FOURTEEN BYTES OF THE FAMILIAR BLOCK MOVE ROUTINE ARE LOADED IN AND THE USR VECTOR SET TO POINT AT THE START OF THIS ROUTINE. THE ROUTINE AS LOADED HAS THE HL REGISTER (SOURCE) SET TO THE START OF THE VIDEO DISPLAY RAM AT 7000H AND WHICH EXTENDS TO 77FFH (THE 2K SCREEN AREA). THE DE REGISTER (DESTINATION) IS FIRST SET TO THE BOTTOM OF BANK 1/2/3 AT C000H. THE SIZE OR BC REGISTER IS SET TO 800H OR 2K. AS THE 2K SCREEN PAGES OCCUR ON SIMPLE BOUNDARIES IN ADDRESS SPACE, (C000H, C8000H, D000H, D8000H ETC.) IT IS ONLY NECESSARY TO ALTER THE MOST SIGNIFICANT BYTE OF THE DESTINATION ADDRESS AS THE 2K SCREEN FILES ARE LOADED FROM DISK. THE ADDRESS OF THIS BYTE IN THE M/L ROUTINE IS ASSIGNED IN LINE 1800 TO ENABLE THE BASIC PROGRAM TO PASS VARIABLES INTO THE M/L ROUTINE.

IT WAS DECIDED THAT THE EASIEST WAY TO PRESENT THE 24 FILENAMES FOR THE SCREENS TO BE PRESENTED TO LOADUP WAS VIA DATA STATEMENTS LOCATED IN LINES 2000 TO 2080. THERE IS SCOPE TO ALTER THIS IF REQUIRED.

CONTINUED NEXT ISSUE . . .

```
10 '*************************
20 '***      SET-UP       ***
30 '*** PROGRAM I OF III  ***
40 '*** TO LOAD 24 HI-RES ***
50 '*** SCREENS INTO 64K  ***
60 '*** MEM. EXP.         ***
70 '***   BOB KITCH  6/88 ***
80 '*************************
98 '
99 '***PUT UP INTRO MESSAGE.
100 GOSUB 620
110 PRINT"THIS IS THE FIRST OF A SERIES OFSUBPROGRAMS DESIGNED";
120 PRINT" TO ALLOW 24HI-RES SCREENS TO BE LOADED AND STORED ";
130 PRINT"IN A 64K MEMORY EXP. THE 2K SCREENS ARE LOADED ";
140 PRINT"VIA DOS.  THE SCREENS ARE STORED IN MEM.  BANKS 1 TO";
150 PRINT" 3 OF THE MEMORY       EXPANSION."
160 GOSUB 600
200 PRINT" <SET-UP> LOWERS TOM TO BFFFH - THE TOP OF BANK 0."
210 PRINT"THE DOS VECTOR IS ALSO LOWERED.":PRINT
220 PRINT" <LOAD-UP> SETS THE BLOCK MOVE  SECTION OF M/L IN ";
230 PRINT"PLACE. ALSO   THE 24 HI-RES SCREENS ARE CALLEDFROM ";
240 PRINT"DATA STATEMENTS. USUALLY   THESE EXIST ON A ";
250 PRINT"<PICTURE DISK> WHILST THE SUBPROGRAMS ARE ON A ";
260 PRINT"<PROGRAM DISK>"
270 GOSUB 600
300 PRINT" <THROW-UP> AUTOMATICALLY PAGES THROUGH THE 24 ";
310 PRINT"SCREENS. THIS IS SIMILAR TO <PICTURE SHOW>."
320 PRINT"THE SPEED AT WHICH THE PAGING   OCCURS CAN BE VARIED."
330 PRINT"HI-SPEED ANIMATION IS QUITE     POSSIBLE."
340 PRINT:PRINT"ALTERNATIVELY, FANCY SCREEN     MOVES CAN BE ";
350 PRINT"SELECTED."
360 GOSUB 600
400 PRINT"THE SUBPROGRAMS CHECK FOR THE   64K MEM. EXP. AND DOS"
410 PRINT:PRINT"TWO A/L PROGRAMS ARE USED."
420 PRINT:PRINT"<START-UP> LOWERS THE DOS       VECTOR."
430 PRINT:PRINT"<MOVE-UP> CONTAINS THE FANCY    SCREEN MOVES."
460 GOSUB 600
498 '
499 '***RUN LOAD-UP PROGRAM.
500 PRINT"LOADING START-UP & LOAD-UP"
510 BRUN"STARTUP"
598 '
599 '***SUBROUTINE FOR NEW SCREEN.
600 PRINT@480,"PRESS ANY KEY TO CONTINUE";
610 A$=INKEY$:A$=INKEY$:IF A$="" THEN GOTO 610
620 CLS:PRINT@13,"SET-UP":PRINT:PRINT:RETURN
999 '
1000 '***CALLING SEQUENCE THROUGH PROGRAMS.
1010 'RUN T:SETUP   CALLS...
1020 '  BRUN B:STARTUP    CALLS...
1030 '    RUN T:LOADUP        CALLS...
1040 '      BLOAD B:MOVEUP
1050 '        RUN T:THROWUP
1060 '
1070 'A/L SOURCE CODE
1080 'SOURCE  W:START   OBJECT  B:STARTUP   ORIGIN  7200H
1090 'SOURCE  W:MOVE    OBJECT  B:MOVEUP    ORIGIN  0BC00H
1099 '
```

. . . . . SET-UP CONTINUED

```
1100 '***LOAD MAP FOR LIVEN-UP.
1110 '0C000H-0FFFFH  - 3 BY 16K BANKS OF PICTURE BUFFERS.
1120 '0BEC8H-0BFFFH  - 310 BYTE DOS VECTOR AND BUFFER.
1130 '0BEABH-0BEB8H  - 14 BYTE G.P. BLOCK MOVE ROUTINE.
1140 '0BC00H-0BE0FH  - 528 BYTE MOVEUP ROUTINES.
1150 '0B200H-0BA00H  - 2K BYTE SCREEN BUFFER.
1160 '0AFFFH         - TOM FOR BASIC PROGRAMS.
1170 '0AFCDH-0AFFFH  - STRING SPACE.
1180 '0AF98H-0AFCCH  - STACK.
1190 ' 849DH-0AF97H  - FREE SPACE.
1200 ' 849DH- 849DH  - DIM VAR. TABLE (NONE).
1210 ' 8481H- 849CH  - VARIABLE TABLE.
1220 ' 7AE9H- 8480H  - PROGRAM STATEMENT TABLE.
1230 ' 7AE9H         - SOB FOR BASIC PROGRAMS.
1240 ' 7200H- 7233H  - STARTUP ROUTINE IN HI-RES SCREEN AREA.
1250 'THE BASIC PROGRAM AREA IS DESCRIBED FOR <THROWUP>.
9998 '
9999 '***UPDATE DISK FILE.
10000 ERA"SETUP":SAVE"SETUP"
20000 END
```

WHAT COMPUTER NOW BY JOE LEON . . .

I GET ASKED THIS QUESTION QUITE OFTEN AND IT'S A VERY DIFFICULT ONE TO ANSWER. IT DEPENDS ON MANY FACTORS WITH THE MOST IMPORTANT TWO BEING COST AND WHAT DO YOU, NOT I WANT TO DO WITH IT. WHAT MAY SUIT ME, MIGHT BE PARTIALLY OR TOTALLY UNSUITABLE FOR YOU.

AS SOME OF YOU ALREADY KNOW AND BY THE TIME YOU READ THIS I SHOULD BE IN POSESSION OF AN IBM AT COMPATIBLE COMPUTER AND I'LL EXPLAIN MY REASONS FOR PURCHASING ABOVE INSTEAD OF AN AMIGA, AMSTRAD, ETC. FOR ME THE CHOICE WAS SIMPLE AS I KNEW WHAT USE IT WOULD BE PUT TO.

NEXT YEAR I'LL BE DOING A COMPUTER OFFICE COURSE TO ENHANCE MY JOB PROSPECTS AND HOPEFULLY LEARN SOME NEW SKILLS AS WELL. MY SON PAUL IS DOING A THREE YEAR COMPUTER PROGRAMMERS COURSE WITH TWO MORE YEARS TO GO. IN BOTH ABOVE AN IBM OR COMPATIBLE IS USED WHICH MADE MY CHOICE EASY AND OTHER FACTORS CAME INTO COSIDERATION AS WELL, EG:-

PRINTERS - MY STAR NX1000 PRINTER IS IBM COMPATIBLE AND WHEN I WAS LOOKING FOR A PRINTER I HAD IBM COMPATIBILITY IN MIND.

PCB CAD & CIRCUIT DESIGN CAD - AS MOST OF YOU ARE AWARE I HAVE A VERY KEEN INTEREST IN HARDWARE MODS TO THE VZ, SO IT GOES WITHOUT SAYING THAT A PCB CAD (PRINTED CIRCUIT BOARD COMPUTER AIDED DESIGN) AND CIRCUIT DESIGN CAD PROGRAM WILL BE OF BENEFIT TO ME AND OTHER VZ USERS AS WELL. AT LONG LAST I LL BE ABLE TO OFFER PCB'S FOR PAST AND FUTURE HARDWARE PROJECTS THAT HAVE APPEARED OR WILL APPEAR IN THE JOURNAL.

ALTHOUGH THE IBM COMPATIBLE COST ME AN ARM AND A LEG I CONSIDER IT AN INVESTMENT IN MY AND MY FAMILY'S FUTURE. YOUR NEEDS MORE THAN LIKELY WILL DIFFER FROM MINE SO TRY TO ENVISAGE YOUR FUTURE AS WELL AS YOUR PRESENT NEEDS AND MAKE YOUR CHOICE ACCORDINGLY.

BACK-UP - THIS IS VERY IMPORTANT. NEXT TIME YOU GO TO A NEWSAGENT HAVE A LOOK AT WHAT MAGAZINES AND HOW MANY ARE AVAILABLE FOR EACH COMPUTER. THIS WILL GIVE A GOOD INDICATION OF SUPPORT FOR A PARTICULAR COMPUTER.

SUITE II CONT. BY R.QUINN

SETTING UP A VPROG IN VIDEO MEMORY

THE MEMORY CELL BEFORE START OF A BASIC PROGRAM MUST CONTAIN A ZERO BYTE, OTHERWISE THE PROGRAM WILL NOT RUN (UNLESS YOU RUN IT FROM A LINE THAT IS NOT THE FIRST LINE OF THE PROGRAM). THAT IS WHY THE LAST BYTE OF THE COMMUNICATIONS REGION (ADDRESS 31464) IS '0'.

SO WE NEED A ZERO BYTE IN ADDRESS 29185, JUST BEFORE START OF A VPROG. WE WILL ALSO NEED ZERO BYTES IN 29186 AND 29187 TO BEGIN WITH, TO INITIALLY SET UP A NULL PROGRAM IN VIDEO MEMORY. ONE EASY WAY TO ZERO ALL OF VIDEO MEMORY FROM 29184 ONWARD IS TO ENTER THE MODE(1) COMMAND. BUT WE'LL DO IT ANOTHER WAY SINCE WE HAVE TO ALTER THE PROGRAM POINTERS ANYWAY:

```
POKE29185,0      <RETURN>
PRINT&           <RETURN>   REM THIS RESETS START POINTER TO 29186
NEW             <RETURN>
```

REM : THIS SETS UP A NULL PROGRAM IN VIDEO MEMORY AND RESETS END POINTER, ETC. TO VIDEO MEMORY (29188)

NOW YOU CAN TYPE IN LINES OF A BASIC PROGRAM AND THE PROGRAM WILL BE PLACED IN VIDEO MEMORY FROM 29186 ON. USE THIS COMMAND FROM TIME TO TIME TO MAKE SURE THE PROGRAM DOES NOT GO BEYOND 30719:

```
PRINTPEEK(30969)+PEEK(30970)*256
```

NOW MAKE A BINARY SAVING OF VIDEO MEMORY FROM 29184 TO 30719 THUS:-

```
BSAVE"YOURNAME",7200,77FF
```

THIS IS YOUR VPROG.

THE ERT FACILITY OF SUITE2 IS ACCESSED USING A PRINT&6 COMMAND.

<CTRL> <P> THEN <SHIFT> <6> THEN <6>

A SERIES OF NUMBERS WILL THEN DISPLAY ON SIX SCREEN LINES:-

```
A
B      C
D      E
F      G
H
I
```

A IS START ADDRESS OF PROGRAM MEMORY.
B IS END ADDRESS OF BASIC PROGRAM IN PROGRAM MEMORY.
C IS LENGTH OF BASIC PROGRAM.
D IS END OF SIMPLE VARIABLES LIST/START OF ARRAY VARIABLES LIST.
E IS LENGTH FROM START OF PROGRAM TO END OF SIMPLE VARIABLES LIST.
F IS END OF ARRAY VARIABLES LIST/START OF FREE MEMORY.
G IS LENGTH FROM START OF PROGRAM TO END OF ARRAY VARIABLES LIST.
H IS ADDRESS OF TOP OF MEMORY [MINUS DISK BUFFER].
I IS THE AMOUNT OF FREE MEMORY AVAILABLE.

UNTIL A BASIC PROGRAM IS RUN, D AND F EQUAL B; E AND G EQUAL C. IF THE PROGRAM DOES NOT USE ARRAYS THEN F EQUALS D AND G EQUALS E.

27/11 KEYBOARDING PART I BY JOE LEON

AS EDITOR I RECEIVE QUITE A BIT OF MAIL RANGING FROM HARD TO READ SCRIBBLED NOTES TO TYPE-WRITTEN LETTERS WITH SPOT ON PUNCTUATION, SPELLING AND LAYOUT AND QUITE A MIXTURE IN BETWEEN. SINCE MY VOLUNTARY EARLY RETIREMENT EARLIER THIS YEAR I'VE TAKEN ADVANTAGE OF ALL THE EXTRA FREE TIME TO LEARN SOME NEW SKILLS BY DOING TWO COURSES.

ONE OF THE COURSES IS A 10 WEEK KEYBOARDING (TYPEWRITING) COURSE TO ENHANCE MY JOB PROSPECTS AND SHOULD ALSO HELP ME TO PREPARE A BETTER PRESENTED PUBLICATION WITH HOPEFULLY LESS ERRORS. SINCE THE 'PUNCTUATION' FOR TYPEWRITING AND WORD-PROCESSING IS SIMILAR I THOUGHT AN ARTICLE ON THE SUBJECT WOULD NOT GO ASTRAY.

I'LL START WITH A BRIEF EXPLANATION OF INTERNATIONAL PAPER SIZES AND THEIR RELATIONSHIP TO EACH OTHER.

A 0 -	1189 X 841MM	A 6 -	148 X 105MM
A 1 -	841 X 594MM	A 7 -	105 X 74MM
A 2 -	594 X 420MM	A 8 -	74 X 52MM
A 3 -	420 X 297MM	A 9 -	52 X 37MM
A 4 -	297 X 210MM	A10 -	37 X 26MM
A 5 -	210 X 148MM		

BY FOLDING ANY SHEET FROM A0 TO A9 ALONG ITS LENGHT AND CUTTING IT IN HALF ALONG THE FOLD WILL PRODUCE TWO SHEETS OF NEXT SMALLER SIZE. EG: AN A3 SHEET WILL PRODUCE TWO A4 SHEETS. MOST OF YOU SHOULD BE FAMILIAR WITH THE A4 SIZE AS IT'S THE ONE USED FOR THE JOURNAL. CONVERSLY TWO A5'S SIDE BY SIDE WILL PRODUCE AN A4 SIZE. TWO TERMS ARE USED TO DESCRIBE HOW INFORMATION IS PRESENTED ON A SHEET OF PAPER WHICH ARE:

PORTRAIT - THIS TERM INDICATES THAT THE SHORT EDGE OF THE PAGE IS AT TOP LIKE IN A 'PORTRAIT' AND THE JOURNAL IS IN THIS STYLE.

LANDSCAPE - THIS TERM INDICATES THAT THE LONG EDGE OF THE PAGE IS AT TOP LIKE IN A 'LANDSCAPE' AND BUSINESS CARDS USE THIS STYLE.

THERE ARE QUITE A FEW STANDARD RULES GOVERNING THE USE OF TYPEWRITTEN OR WORDPROCESSOR GENERATED TEXT WHICH ALSO APPLY TO ADDRESSING ENVELOPES AS WELL. IN DUE COURSE I'LL COVER MOST OF THEM IN THIS AND COMING ISSUES. THE A4 SIZE WILL BE USED AS REFERENCE IN THE SERIES.

LINE SPACING - S/S, D/S AND T/S:

S/S - (SINGLE LINE SPACING) - THIS TERM INDICATES THAT THERE ARE NO CLEAR LINES BETWEEN LINES OF TEXT AS IN THIS PARARAGRAPH.

D/S - (DOUBLE LINE SPACING) - THIS TERM INDICATES THAT THERE IS ONE CLEAR LINE BETWEEN LINES OF TEXT AS IN THIS PARARAGRAPH.

T/S - (TRIPLE LINE SPACING) - THIS TERM INDICATES THAT THERE ARE TWO CLEAR LINES BETWEEN LINES OF TEXT AS IN THIS PARAGRAPH.

KEYBOARDING PART I CONTINUED

LINES PER PAGE - 70 LINES WILL FIT ON AN A4 SIZE PAGE BUT IN PRACTICE YOU'LL ONLY GET ABOUT 58 LINES. THE REASON FOR THAT IS FAIRLY SIMPLE. WHEN YOU FEED A SHEET OF PAPER IN TO YOUR TYPEWRITER OR PRINTER YOU'LL FIND THAT ABOUT SIX LINES AT TOP AND BOTTOM CANNOT BE PRINTED ON. MOST TYPEWRITERS ARE DESIGNED THAT WAY TO LEAVE A CLEAR TOP AND BOTTOM MARGIN BUT THE NUMBER OF CLEAR LINES TOP AND BOTTOM COULD VARY ON PRINTERS.

SPEECH SYNTHESISER BY GARY BULLEY

I HAVE JUST COMPLETED THE CONSTRUCTION OF A TEXT TO SPEECH SYNTHESISER AND FOUND THAT WHEN I CONNECT IT TO THE PRINTER PORT OF THE VZ AND USE THE LPRINT COMMAND SOME CHARACTERS ARE LOST DUE TO THE SLOW INPUT SPEED OF THE SYNTHESISER. TO OVERCOME THIS PROBLEM I WROTE THE SMALL PROGRAM LISTED BELOW AND THOUGHT IT MAY BE OF INTEREST TO OTHER READERS.

```
10 CLS:POKE30862,80:POKE30863,52
20 POKE(PEEK(30753)*256+PEEK(30752)),124
30 IFINKEY$=S$THEN30
40 S$=INKEY$:S$=INKEY$
50 IFS$=""THEN40
60 X=USER(X)
70 PRINTS$;
80 S=ASC(S$):OUT0,S:OUT1,0
90 GOTO 20
```

A DESCRIPTION OF THE PROGRAM IS AS FOLLOWS :-

LINE 10 CLEARS SCREEN AND POKES THE ADDRESS OF THE ROM 'BEEP' ROUTINE.

LINE 20 PEEKS THE CURRENT CURSOR POSITION (30752 30753) AND POKES THE CURSOR CHARACTER INTO THIS POSITION.

NOTE: READERS MAY WISH TO USE THEIR OWN CURSOR CHARACTER HERE. SIMPLY REPLACE THE 124 WITH CURSOR CHARACTER OF YOUR OWN CHOICE.

LINE 30 HOLDS THE PROGRAM UNTIL COMPLETION OF A KEYSTROKE.

LINES 40-50 INPUTS A CHARACTER FROM THE KEYBOARD.

LINE 60 SOUNDS A BEEP AFTER A KEYPRESS.

LINE 70 PRINTS A CHARACTER ON THE SCREEN.

LINE 80 OUTPUTS AN ASCII CHARACTER TO THE PRINTER PORT AND ACTIVATES THE STROBE LINE.

LINE 90 RETURNS TO LINE 20 AND THE PROGRAM IS READY FOR MORE TEXT.

ON THE NEXT PAGE IS A DEMONSTRATION PROGRAM INCORPORATING ABOVE ROUTINE. NEEDLESS TO SAY PROGRAM COULD BE ENLARGED AND TAKEN MUCH FURTHER AND I LEAVE IT TO YOUR IMAGINATION. HAVE FUN.

```
10 POKE30862,80:POKE30863,52:GOTO 500
20 POKE(PEEK(30753)*256+PEEK(30752)),124
30 IF INKEY$=S$ THEN 30
40 S$=INKEY$:S$=INKEY$
50 IF S$=""THEN 30 ELSE IF S$="^"THEN 500:REM ^ = SHIFT+N
60 X=USR(X)
70 PRINT S$;
80 S=ASC(S$):OUT0,S:OUT1,0
90 GOTO 20
95 :
100 DIMS$(100)
110 S$="MACHINE GUN.GGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGGG"
120 PRINT@74,"MACHINE GUN"
130 TD=2600:GOSUB 400
140 SS$="HELICOPTER.QQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQQ"
150 PRINT@138,"HELICOPTER"
160 TD=4000:GOSUB 400
170 SS$="FROG.BHBHBHBH.BHBHBHBH,BHBHBHBH..BHBHBH.BHBHBHBH,BHBHB"
180 PRINT@202,"FROG"
190 TD=2700:GOSUB 400
200 PRINT@266,"SPRING"
210 SS$="SPRING.HNKNKNKNKNKNKNKNKNKNKNKNKNKNKNKNKNH"
220 TD=5000:GOSUB 400
230 PRINT@330,"RAYGUN"
240 SS$="RAYGUN.JHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJHJ"
250 TD=2800:GOSUB 400
260 GOTO 500
390 :
400 S=LEN(SS$)
410 FORI=1 TO S
420 OUT 0,ASC(MID$(SS$,I,1)):OUT 1,0
430 NEXT
440 OUT 0,ASC(CHR$(13)):OUT 1,0
450 FOR I=1 TO TD:NEXT
460 RETURN
490 :
500 CLS:PRINT@35,"SPEECH  SYNTHESISER  DEMO"
510 PRINT@67,"-----------------------"
520 PRINT@166,"[D] - DEMONSTRATION"
530 PRINT@262,"[E] - ENTER TEXT"
540 PRINT@358,"[^] - RETURN TO MENU":REM ^ = SHIFT+N
550 PRINT@396,"FROM TEXT ENTRY":SOUND 25,3
560 :
570 A$=INKEY$:A$=INKEY$:IFA$=""THEN 570 ELSE X=USR(X)
580 IF A$="D" THEN CLS:RUN 100
590 IF A$="E" THEN CLS:GOTO 20 ELSE 570
```

NOTE 1: THE TD (TIME DELAY) IN LINES 130, 160, 190, 220, 250 AND EXECUTED BY LINE 450 MAY NEED SOME ADJUSTING. THE IDEA IS TO LEAVE A SHORT PAUSE BETWEEN THE END OF ONE SPOKEN WORD/SOUND DEMO AND THE START OF THE NEXT ONE. THE ONES IN ABOVE LINES ARE A GUIDE ONLY.

NOTE 2: SOME OF THE ROUTINES MAY NOT WORK WITH YOUR SYNTHESISER AS THEY DEPEND TO A LARGE EXTENT ON YOUR PRINTER INTERFACE. WE TRIED THREE PRINTER INTERFACES AND EACH PRODUCED A DIFFERENT RESULT. SOME EXPERIMENTATION MAY BE NECESSARY.

CHECK DISK BY DAVE MITCHELL 27/14

```
00001 ;CHECK DISK ROUTINE
00002 ; ORIGIN 9000 HEX
00003        CALL 01C9H
00004        LD   HL,MES1
00005        CALL 2B75H
00006 A1     CALL 0049H
00007        CP   0DH
00008        JR   NZ,A1
00009        LD   (IY+11H),0
00010        LD   (IY+12H),0
00011        DI
00012        CALL 4008H
00013        LD   BC,0032H
00014        CALL 4038H
00015 A2     CALL 4035H
00016        OR   A
00017        JR   Z,A3
00018        LD   HL,MES2
00019        JP   END
00020 A3     INC  (IY+11H)
00021        LD   A,0FH
00022        CP   (IY+11H)
00023        JR   NZ,A2
00024        CALL 4035H
00025        OR   A
00026        JP   NZ,S1
00027        LD   (IY+11H),0
00028        LD   (IY+12H),1
00029        LD   L,(IY+31H)
00030        LD   H,(IY+32H)
00031        LD   E,(IY+34H)
00032        LD   D,(IY+35H)
00033        LD   BC,0050H
00034        LDIR
00035 A4     DI
00036        CALL 4035H
00037        OR   A
00038        JP   NZ,ERR
00039 A4A    INC  (IY+11H)
00040        LD   A,10H
00041        CP   (IY+11H)
00042        JR   NZ,A4
00043        LD   (IY+11H),0
00044        INC  (IY+12H)
00045        LD   A,28H
00046        CP   (IY+12H)
00047        JR   NZ,A4
00048        LD   L,(IY+31H)
00049        LD   H,(IY+32H)
00050        PUSH HL
00051        LD   (HL),0
00052        POP  DE
00053        PUSH DE
00054        INC  DE
00055        LD   BC,007FH
00056        LDIR
00057        POP  DE
00058        LD   L,(IY+34H)
00059        LD   H,(IY+35H)
00060        LD   BC,0050H
00061        LDIR
00062        LD   (IY+11H),0FH
00063        LD   (IY+12H),0
00064        CALL 4032H
00065        OR   A
00066        JR   Z,A5
00067 S1     LD   HL,MES3
00068        JP   END
00069 A5     LD   L,(IY+31H)
00070        LD   H,(IY+32H)
00071        LD   DE,0
00072        LD   C,4EH
00073 A6     LD   B,8
00074        LD   A,(HL)
00075 A7     RRC  A
00076        JR   C,A8
00077        INC  DE
00078 A8     DJNZ A7
00079        INC  HL
00080        DEC  C
00081        JR   NZ,A6
00082        PUSH DE
00083        PUSH DE
00084        POP  HL
00085        CALL 0FAFH
00086        LD   HL,MES4
00087        CALL 2B75H
00088        POP  HL
00089        PUSH HL
00090        SRL  H
00091        RR   L
00092        SRL  H
00093        RR   L
00094        SRL  H
00095        RR   L
00096        CALL 0FAFH
00097        LD   A,2EH
00098        CALL 033AH
00099        POP  HL
00100        LD   A,7
00101        AND  L
00102        INC  A
00103        LD   B,A
00104        LD   HL,0FF83H
00105        LD   DE,007DH
00106 A9     ADD  HL,DE
00107        DJNZ A9
00108        CALL 0FAFH
00109        LD   HL,MES5
00110 END    CALL 2B75H
00111        CALL 400BH
00112        JP   1A19H
00113 ERR    LD   L,(IY+34H)
00114        LD   H,(IY+35H)
00115        LD   A,(IY+12H)
00116        DEC  A
00117        SLA  A
00118        LD   E,A
00119        LD   D,0
00120        LD   A,(IY+11H)
```

27/15 . . . CHECK DISK CONTINUED

```
00121        CP    8
00122        CCF
00123        ADC   HL,DE
00124        AND   7
00125        INC   A
00126        LD    B,A
00127        LD    C,(HL)
00128        RLC   C
00129 A13    RRC   C
00130        DJNZ  A13
00131        SET   0,C
00132        LD    B,A
00133        RRC   C
00134 A14    RLC   C
00135        DJNZ  A14
00136        LD    (HL),C
00137        JP    A4A
00138 MES1   DEFB  1FH
00139 *         CHECK DISK*
00140        DEFB  0DH
00141 *      WRITTEN BY D.MITCHELL*
00142        DEFB 0DH
00143 *    WHEN READY PRESS RETURN
00144        DEFB 0DH
00145        NOP
00146 MES2 EQU   $
00147 *ERROR IN DIRECTORY SECTORS
00148 * TRY  REFORMATTING*
00149        DEFW 000DH
00150 MES3 EQU   $
00151 *ERROR IN STATUS SECTOR TRY
00152 *        REFORMATTING*
00153        DEFW 000DH
00154 MES4 EQU   $
00155 * SECTORS FREE    *
00156        NOP
00157 MES5 EQU   $
00158 *K FREE*
00159        DEFW 0D0DH
00160        NOP
```

CHECK DISK OPERATION EXPLAINED BY DAVE MITCHELL

I WILL ATTEMPT TO EXPLAIN HOW "CHECK DISK" (CHKDSK) OPERATES.

AFTER A DISK IS INITIALIZED OR FORMATTED THE DRIVE HEAD IS MOVED TO TRACK ZERO. THE DOS THEN READS THE IDENTIFICATION ADDRESS MARK AND COMPARES THE TRACK AND SECTOR NUMBERS WITH WHAT IS IN THE DOS COMMUNICATION RAM IF ALL IS OK THEN THE NEXT SECTOR NUMBER IS CHECKED. THIS CONTINUES UNTIL ALL SECTORS ARE CHECKED AND THE DRIVE HEAD IS MOVED TO THE NEXT TRACK AND THE PROCESS IS STARTED OVER AGAIN. THIS IS DONE UNTIL ALL TRACKS ARE CHECKED OR IF A SECTOR NUMBER IS NOT FOUND THEN THE INPUT/OUTPUT ERROR IS PRINTED.

GREAT, BUT ONLY THE IDENTIFICATION ADDRESS MARK WAS CHECKED WHAT ABOUT THE REST OF THE FORMAT? SORRY NO CHECKS ARE DONE AND THIS IS HOW THE I/O ERRORS HAPPEN WHEN SAVING AND LOADING FROM DISK.

CHECK DISK READS THE WHOLE FORMAT AS IF IT WAS LOADING A PROGRAM FROM DISK AND IF A SECTOR HAS AN ERROR THEN THAT SECTOR IS WRITTEN TO THE TRACK MAP AS BEING USED. BY LOCKING OFF THE REJECTED SECTORS FROM USE HOPEFULLY WE CAN SLOW THE I/O ERRORS DOWN. D'ONT HOLD YOUR BREATH WHILE CHECK DISK IS OPERATING AS IT TAKES QUITE SOME TIME AND WILL TAKE LONGER THE MORE SECTORS CANNOT BE FOUND.

LETS PULL THE SOURCE CODE APART . . . . .

```
00001 ;CHECK DISK ROUTINE
00002 ;SELECT DRIVE
00004       CALL 01C9H
00005       LD   HL,MES1
00006       CALL 2B75H
```

LINES 4,5 & 6 CLEARS THE SCREEN AND PRINTS THE SIGN ON MESSAGE (MES1).

```
00007 A1    CALL 0049H
```

LINE 7 IS UNUSUAL AS NOT MANY PEOPLE USE THIS CALL. IT SCANS THE KEYS BUT DOES NOT RETURN TO THE CALLER UNTIL A KEY IS PRESSED.

```
00008          CP    0DH
00009          JR    Z,A1A
00010          CP    32H
00011          JR    Z,TWO
00012          CP    31H
00013          JR    NZ,A1
00014          LD    A,10H
00015          JR    T1
00016 TWO      LD    A,80H
00017 T1       PUSH  AF
00018          DI
00019          CALL  4008H
00020          LD    A,(IY+20)
00021          OR    A
00022          JR    Z,T2
00023          LD    B,A
00024          CALL  403EH
00025 T2       CALL  400BH
00026          POP   AF
00027          LD    (IY+0BH),A
```

LINES 8 TO 27 THIS IS MY WAY OF STOPPING THE DRIVES FROM BASHING. IF YOU SELECT THE OTHER DRIVE AND THE HEAD IS IN A DIFFERENT LOCATION, DOS WILL RESET THE HEAD BUT BY RESETTING THE HEADS TO TRACK ZERO , I AVOID THIS PROBLEM.

```
00028 A1A      LD    (IY+11H),0
00029          LD    (IY+12H),0
00030          DI
00031          CALL  3450H
00032          CALL  4008H
00033          LD    BC,0032H
00034          CALL  4038H
00035 A2       CALL  4035H
00036          OR    A
00037          JR    Z,A3
00038          LD    HL,MES2
00039          JP    END
```

LINES 28 TO 39 CHECKS THE DIRECTORY FOR AN ERROR . IF FOUND THE PROGRAM PRINTS MESSAGE 2 (MES2) AND JUMPS TO BASIC.

```
00040 A3       INC   (IY+11H)
00041          LD    A,0FH
00042          CP    (IY+11H)
00043          JR    NZ.A2
00044          CALL  4035H
00045          OR    A
00046          JP    NZ.S1
```

LINES 40 TO 46 CHECK THE TRACK MAP SECTOR OR AS YOU KNOW IT THE STATUS.

AGAIN IF AN ERROR IS FOUND THE PROGRAM IS DIRVERTED TO 'S1' WHICH PRINTS MESSAGE 3 (MES3) AND JUMPS TO BASIC.

```
00047          LD    (IY+11H),0
00048          LD    (IY+12H),1
00049          LD    L,(IY+31H)
00050          LD    H,(IY+32H)
00051          LD    E,(IY+34H)
00052          LD    D,(IY+35H)
00053          LD    BC,0050H
00054          LDIR
```

LINES 49 TO 54 MOVES THE TRACK MAP INTO THE MAP BUFFER SO IT IS NOT OVER WRITTEN.

```
00055 A4       DI
00056          CALL  4035H
00057          OR    A
00058          JP    NZ,ERR
00059 A4A      INC   (IY+11H)
00060          LD    A,10H
00061          CP    (IY+11H)
00062          JR    NZ,A4
00063          LD    (IY+11H),0
00064          INC   (IY+12H)
00065          LD    A,28H
00066          CP    (IY+12H)
00067          JR    NZ,A4
```

LINES 55 TO 67, THIS IS THE MAIN PART OF THE PROGRAM. EVERY TRACK/SECTOR FROM TRACK ONE TO THIRTY NINE IS READ INTO MEMORY. LINES 57 & 58 ARE THE ERROR CAPTURE THE OR A TESTS THE A REGISTER AND SETS THE ZERO FLAG AND LINE 58 JUMPS TO THE ERROR (ERR) ROUTINE IF IT IS NOT ZERO (NZ).

```
00068          LD    L,(IY+31H)
00069          LD    H,(IY+32H)
00070          PUSH  HL
00071          LD    (HL),0
00072          POP   DE
00073          PUSH  DE
00074          INC   DE
00075          LD    BC,007FH
00076          LDIR
00077          POP   DE
00078          LD    L,(IY+34H)
00079          LD    H,(IY+35H)
00080          LD    BC,0050H
00081          LDIR
```

27/17 CHECK DISK EXPLANATION CONT.

LINE 68 TO 81 CLEARS THE DATA BUFFER AND MOVES THE TRACK MAP INTO THE DATA BUFFER SO IT CAN BE SAVED TO DISK.

```
00082         CALL WP
00083         LD   (IY+11H),0FH
00084         LD   (IY+12H),0
00085         CALL 4032H
00086         OR   A
00087         JR   Z,A5
00088 S1      LD   HL.MES3
00089         JP   END
```

LINES 82 TO 89 CHECKS FOR WRITE PROTECT AND WRITES TO DISK, IF ALL WELL MESSAGE 3 (MES3) WON'T BE PRINTED.

```
00090 A5      LD   L,(IY+31H)
00091         LD   H,(IY+32H)
00092         LD   DE,0
00093         LD   C,4EH
00094 A6      LD   B,8
00095         LD   A,(HL)
00096 A7      RRC  A
00097         JR   C,A8
00098         INC  DE
00099 A8      DJNZ A7
00100         INC  HL
00101         DEC  C
00102         JR   NZ,A6
00103         PUSH DE
00104         PUSH DE
00105         POP  HL
00106         CALL 0FAFH
00107         LD   HL,MES4
00108         CALL 2B75H
00109         POP  HL
00110         PUSH HL
00111         SRL  H
00112         RR   L
00113         SRL  H
00114         RR   L
00115         SRL  H
00116         RR   L
00117         CALL 0FAFH
00118         LD   A,2EH
00119         CALL 033AH
00120         POP  HL
00121         LD   A,7
00122         AND  L
00123         INC  A
00124         LD   B,A
00125         LD   HL,0FF83H
00126         LD   DE.007DH
00127 A9      ADD  HL,DE
00128         DJNZ A9
00129         CALL 0FAFH
00130         LD   HL,MES5
00131 END     CALL 2B75H
00132         CALL 400BH
00133         JP   1A19H
```

LINES 90 TO 133 WORKS OUT AND PRINTS THE STATUS TO THE SCREEN TURNS OFF THE DRIVE AND JUMPS TO BASIC.

```
00134 ERR     LD   L,(IY+34H)
00135         LD   H,(IY+35H)
00136         LD   A,(IY+12H)
00137         DEC  A
00138         SLA  A
00139         LD   E,A
00140         LD   D.0
00141         LD   A,(IY+11H)
00142         CP   8
00143         CCF
00144         ADC  HL,DE
00145         AND  7
00146         INC  A
00147         LD   B,A
00148         LD   C,(HL)
00149         RLC  C
00150 A13     RRC  C
00151         DJNZ A13
00152         SET  0,C
00153         LD   B,A
00154         RRC  C
00155 A14     RLC  C
00156         DJNZ A14
00157         LD   (HL),C
00158         JP   A4A
```

LINES 134 TO 158. THIS IS THE ERROR (ERR) TRAP ROUTINE BUT ACTUALLY IT IS THE ROUTINE THAT WRITES TO THE TRACK MAP BUFFER AND WORKS OUT THE TRACK/SECTOR THAT HAS BEEN USED.

```
00159 WP      IN   A,(13H)
00160         BIT  7,A
00161         RET  Z
00162         CALL 400BH
00163         EI
00164         LD   HL,WP1
00165         CALL 2B75H
00166         LD   HL,WP2
00167         CALL 2B75H
00168 FA      LD   A,(7AAFH)
00169         OR   A
00170         JR   NZ.FA
00171 K2      CALL 0049H
00172         CP   0DH
00173         JR   NZ,K2
00174         DI
00175         CALL 3450H
```

CHECK DISK EXPLANATION CONT.

```
00176      CALL 4008H
00177      LD   BC,0032H
00178      CALL 4038H
00179      RET
```

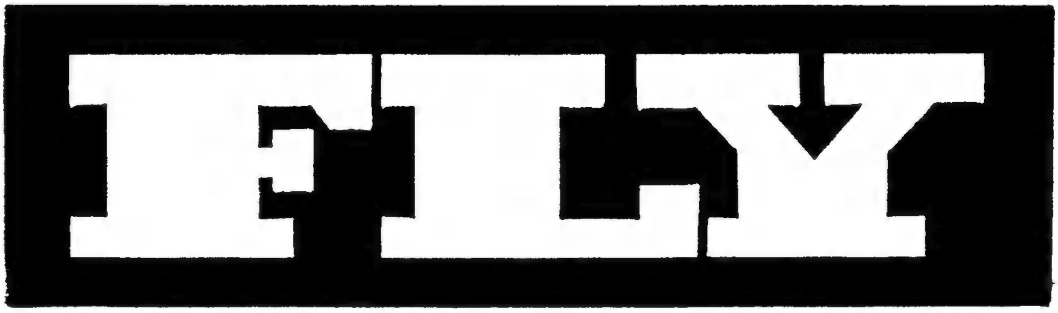
LINES 159 TO 179 ARE THE WRITE PROTECT ROUTINE, IF THE DISK IS WRITE PROTECTED THE PROGRAM WAITS FOR THE USER TO REMOVE THE WRITE PROTECT LABEL AND PRESS THE RETURN KEY. YOU WILL ALSO NOTICE I HAVE MADE USE OF THE ROUTINE AT 0049 HEX AGAIN. IF YOU DON'T REMOVE THE WRITE PROTECT LABEL AND PRESS THE RETURN KEY, THE PROGRAM WILL GO THROUGH THE PROCEDURE OF SAVING THE TRACK MAP TO DISK BUT IT WILL NOT HAPPEN DUE TO THE HARDWARE OR CIRCUIT OF THE DRIVE.

```
00180 MES1 EQU $
00181 *         CHECK DISK*
00182      DEFB 0DH
00183 *    WRITTEN BY D.MITCHELL*
00184 WP2  DEFB 0DH
00185 *   WHEN READY PRESS RETURN
00186      DEFB 0DH
00187      NOP
00188 MES2 EQU  $
00189 *ERROR IN DIRECTORY SECTORS
00190 * TRY  REFORMATTING*
00191      DEFW 000DH
00192 MES3 EQU  $
00193 *ERROR IN STATUS SECTOR TRY
00194 *       REFORMATTING*
00195      DEFW 000DH
00196 MES4 EQU  $
00197 * SECTORS FREE   *
00198      NOP
00199 MES5 EQU  $
00200 *K FREE*
00201      DEFW 0D0DH
00202      NOP
00203 WP1  EQU  $
00204      DEFB 0DH
00205 *REMOVE WRITE PROTECT LABEL*
00206      NOP
```

LINES 180 TO 206 ARE ALL THE MESSAGES USED IN CHECK DISK. I HOPE I HAVE SHONE AT LEAST A DIM LIGHT ONTO HOW THE PROGRAM WORKS. DAVE.

GRAPHIC PUZZLE REVEALED . . .

SOME OF OUR READERS GOT IT STRAIGHT AWAY, OTHERS TOOK A BIT LONGER WHILE THERE COULD BE SOME STILL WORKING ON IT. ONE GLANCE BELOW AND ALL SHOULD BE CLEAR. WHITE LETTERS ON A WHITE BACKGROUND MISLED SOME OF YOU.



CONTRIBUTIONS TO THE HUNTER VALLEY VZ JOURNAL :-

IF YOU ARE THINKING OF CONTRIBUTING TO THE JOURNAL THE PREFERED FORMAT IS BASIC LISTINGS, WORD PROCESSOR OR SOURCE CODE FILES ON TAPE OR DISK. FILES FROM THE FOLLOWING WORD PROCESSORS CAN BE ACCEPTED :-

E & F TAPE OR DISK PATCH 3.1-3.3, WORDPRO CARTRIDGE, WORDPRO PATCH AND ALL QUICKWRITE WORD PROCESSOR FILES.

WANTED TO BUY --------------------

64K RAM PACKS & VZ200 6K RAM BOARDS - CONTACT JOE LEON
22 DRURY STREET WALLSEND NSW 2287 --- PHONE (049) 51 2756

CLUB MEETINGS -- ALL WELCOME --

FIRST FRIDAY OF MONTH - NO MEETING IN JANUARY 1990

VENUE - JESMOND NEIGHBOURHOOD CENTRE MORDUE PARADE JESMOND
(REAR STOCKLAND MALL - BIG W)

FEBRUARY 2 - MODEM & SERIAL INTERFACE DEMO BY PETER HICKMAN
MARCH 2 - IBM PC COMPATIBLE
APRIL 6 - VZ MOUSE BY GARY BULLEY

FUTURE DEMONSTRATIONS -

EPROM PROGRAMMER & ERASER, AUCTION NIGHT - USING THE VZ, RITTY, ETC.
IF YOU HAVE ANY IDEAS FOR A DEMONSTRATION OR A SUBJECT THEN PLEASE LET YOUR COMMITTEE KNOW.

CLUB COMMITTEE & SUBSCRIPTIONS -

PRESIDENT --------- ROSS WOODS --- (049) 71 2843
SECRETARY/EDITOR -- JOE LEON ----- (049) 51 2756
TREASURER --------- GARY BULLEY -- (049) 54 7561
COMMITTEE MEMBERS - COLIN BRIDGE - PETER JONES

SUBSCRIPTION TO - AUST. - 6 MONTHS $11.00 - 12 MONTHS $21.00
H.V.VZ.JOURNAL - N. Z. - 6 MONTHS $13.00 - 12 MONTHS $26.00

HUNTER VALLEY VZ USERS' GROUP - PO BOX 161 JESMOND 2299
NEW SOUTH WALES AUSTRALIA

VZ USER GROUPS & PUBLICATIONS --

J.C.E. D'ALTON 39 AGNES STREET TOOWONG QUEENSLAND 4066
LE'VZ OOP (VZ MAGAZINE) - VSOFTWAREZ/SOFTWARE/HARDWARE FOR SALE

VZ DOWN UNDER - VZ MAGAZINE - 6 ISSUES - $18.00 PER YEAR
HARRY HUGGINS 12 THOMAS SREET MITCHAM VICTORIA 3132

WAVZ - WESTERN AUSTRALIA VZ USER GROUP
GRAEME BYWATER P O BOX 388 MORLEY W A 6062

BRISBANE VZ USERS WORKSHOP - C/O 63 TINGALPA ST. WYNUM WEST 4178
SOFTWARE FOR SALE - DISK MENU

NOTE :- WHEN WRITING TO ANY ABOVE OR H.V.VZ. USERS' GROUP FOR INFORMATION PLEASE ENCLOSE A S.S.A.E. OR NZ 2 INT. REPLY COUPONS.

FOR SALE E & F W.P.PATCH 3.3

PATCH 3.3 WRITTEN BY DAVE MITCHELL WILL CONVERT YOUR E & F TAPE WORD PROCESSOR FOR FULL DISK USE WHILE RETAINING ALL ORIGINAL FUNCTIONS. BELOW ARE ADDED DISK COMMANDS & FUNCTIONS :-

LOAD, SAVE, ERASE, RENAME, DIRECTORY, INITIALIZE, UPDATE, DRIVE 1 & 2, SHIFTLOCK & IMBEDDED PRINTER CONTROL CODES PLUS CTRL+P WHICH BYPASSES PRINT MENU AND PRINTS TO SCREEN OR PRINTER. A ROUTINE IS ALSO PROVIDED TO CONVERT YOUR BASIC PROGRAM OR SOURCE CODE FILES INTO WORD PROCESSOR FILES.

PATCH 3.3 HAS PROVISION FOR IMBEDDING PRINTER CONTROL CODES IN TEXT AND FAST SAVING AND LOADING OF TEXT DATA TO AND FROM DISK USING BLOCK SAVE/LOAD TECHNIQUES. PRINTER CONTROL CODES CAN BE SAVED TO TAPE OR DISK.

BSTWP.F - THIS UTILITY PROVIDED WITH PATCH 3.3 WILL CONVERT BASIC PROGRAMS AND ED/ASS. SOURCE CODE FILES INTO WORD PROCESSOR FILES.

SYSTEM REQUIREMENTS - VZ 300 + 16K RAM PACK - VZ 200 + 26K

PATCH 3.3 IS COPYRIGHT TO AND ONLY AVAILABLE FROM :-
HUNTER VALLEY VZ USERS' GROUP P.O.BOX 161 JESMOND 2299
N.S.W. AUSTRALIA - PHONE JOE LEON (049) 51 2756

PRICE - AUS/NZ AU$20.00 - UPDATE - AUS-$10.00 - NZ-AU$11.00.
UPDATING AVAILABLE ONLY TO PREVIOUS PURCHASERS OF PATCHES.

FOR MORE INFORMATION WRITE TO H.V.VZ.U.G. ENCLOSING A SSAE.

EXTENDED DOS V1.3 - $15.00

UPDATED VERSION WITH EXTRA COMMANDS ADDED :-

OLD COMMANDS - MERGE, DIRA, LDIRA, DIRB, LDIRB, OLD, OLD., DEC, HEX, STATUSA AND LSTATUSA. STATUSA AND LSTATUSA ALSO WORKS WITH VERSION 1.0 DOS.

NEW COMMANDS :-

MENU - LOADS AND RUNS BINARY OR TEXT MENU PROGRAM FROM DISK.
CODE - SIMPLIFIES USING PRINTER CONTROL CODES DIRECTLY OR FROM WITHIN A PROGRAM.
LTAB - IS FOR SETTING OF LEFT MARGIN.
MOVE - MOVES BASIC FILE FROM DISK TO CHOSEN MEMORY ADDRESS.
UPD - ERASES OLD FILE AND SAVES WITH SAME FILE NAME.

MENU/FILE COPIER - $15.00

THIS UTILITY WILL READ YOUR DISK DIRECTORY AND PRESENT YOU WITH SEVERAL OPTIONS. USING THE CURSOR YOU CAN RUN/BRUN ANY PROGRAM OR SELECT FILE COPY, REN, ERASE, DRIVE 1 OR 2, ETC. BESIDES COPYING TEXT AND BINARY FILES ALL OTHER FILES CAN BE COPIED AS WELL EXEPT FOR DATA FILES.

FOR PURCHASE OR INFORMATION CONTACT DAVE MITCHELL - (079) 27 8519
24 ELPHINSTONE ST. NORTH ROCKHAMPTON QUEENSLAND 4701

FOR INFORMATION OR DEMONSTRATION IN NEWCASTLE AREA CONTACT :-
JOE LEON - (049) 51 2756 - 22 DRURY STREET WALLSEND NSW 2287

TEST SUBROUTINE FOR PRESENCE OF CODE SEQUENCE--FOUR BYTES

```
 1700  6A4   3E         62
LD          A,#AA
 1701         AA        170          170 IS THE FIRST TEST BYTE
--------------------------------
 1702  6A6   BE         190          IS BYTE AT ADDRESS IN HL REGISTERS = 170?
CP          (HL)
--------------------------------
 1703  6A7   23         35           NEXT ADDRESS
INC         HL
--------------------------------
 1704  6A8   C0         192          RETURN IF BYTE IS NOT '170'
RET         NZ
--------------------------------
 1705  6A9   2F         47           ELSE COMPLEMENT BYTE IN ACCUMULATOR
CPL
--------------------------------
 1706  6AA   BE         190          IS BYTE AT SECOND ADDRESS IN HL REGISTERS = NEW BYTE IN ACCUMULATOR?
CP          (HL)
--------------------------------
 1707  6AB   23         35           NEXT ADDRESS
INC         HL
--------------------------------
 1708  6AC   C0         192          RETURN IF BYTES ARE NOT EQUAL
RET         NZ
--------------------------------
 1709  6AD   3E         62           '231' IS THE THIRD TEST BYTE
LD          A,#E7
 1710         E7        231
--------------------------------
 1711  6AF   BE         190          IS BYTE AT THIRD ADDRESS IN HL REGISTERS = '231'?
CP          (HL)
--------------------------------
 1712  6B0   23         35           FOURTH AND FINAL ADDRESS
INC         HL
--------------------------------
 1713  6B1   C0         192          RETURN IF BYTES ARE NOT EQUAL
RET         NZ
--------------------------------
 1714  6B2   2F         47           ELSE COMPLEMENT BYTE IN ACCUMULATOR
CPL
--------------------------------
 1715  6B3   BE         190          AND COMPARE IT WITH BYTE AT ADDRESS IN HL REGISTERS
CP          (HL)
--------------------------------
 1716  6B4   23         35           ADDRESS IN HL REGISTERS IS NOW FIRST ADDRESS BEYOND FOUR TEST LOCATIONS
INC         HL
--------------------------------
 1717  6B5   C0         192          RETURN IF TWO BYTES ARE NOT EQUAL
RET         NZ
--------------------------------
 1718  6B6   FB         251          ELSE ENABLE INTERRUPTS
EI
--------------------------------
 1719  6B7   E9         233          AND JUMP TO ADDRESS IN HL REGISTERS--THE IPL SEQUENCE IS TERMINATED
JP          (HL)
--------------------------------
```

IPL SEQUENCE DECODING CONTINUED . . 27/25

```
CALLED FROM #F9

--------------------------------
 6989  1B4D      2A          42   #78A4 = 30884:
LD             HL,(#78A4)
 6990            A4         164   GET START OF PROGRAM
 6991            78         120   FROM COM. REGION POINTER
--------------------------------
 6992  1B50      CD         205
CALL           #1DF8
 6993            F8         248   #1DF8 = 7672:
 6994            1D          29   TO TURN TRACE OFF
================================
 6995  1B53      32          50
LD             (#78E1),A
 6996            E1         225   #78E1 = 30945: ZERO THE
 6997            78         120   AUTO INPUT FLAG--NO AUTO
--------------------------------
 6998  1B56      77         119   ZERO THE FIRST TWO CELLS
LD             (HL),A
--------------------------------
 6999  1B57      23          35   OF PROGRAM MEMORY--31465
INC            HL
--------------------------------
 7000  1B58      77         119   AND 31466, THEN
LD             (HL),A
--------------------------------
 7001  1B59      23          35   SET POINTER TO START OF
INC            HL
--------------------------------
 7002  1B5A      22          34   VARIABLE LIST TABLE --
LD             (#78F9),HL
 7003            F9         249   #78F9 = 30969: TO 31467,
 7004            78         120   AS END OF PROGRAM
--------------------------------
 7005  1B5D      2A          42   RECOVER START OF (NULL)
LD             HL,(#78A4)
 7006            A4         164   PROGRAM FROM POINTER
 7007            78         120   #78A4 = 30884
--------------------------------
 7008  1B60      2B          43   BACKSPACE TO 31464--LAST
DEC            HL
--------------------------------
 7009  1B61      22          34   LOCATION IN COM. REGION
LD             (#78DF),HL
 7010            DF         223   #78DF/E0 IS POINTER TO
 7011            78         120   PROGRAM START WHEN RUN
--------------------------------
 7012  1B64      06           6   LOAD 26 CELLS IN COM.RES.
LD             B,#1A
 7013            1A          26   FOR VARIABLE DECLARATION
--------------------------------
 7014  1B66      21          33   TABLE, STARTING AT
LD             HL,#7901
 7015            01           1   #7901 = 30977: WITH '4'
 7016            79         121   CODE FOR SINGLE PRECISION
--------------------------------
 7017  1B69      36          54   VARIABLE DECLARATION
LD             (HL),#04
 7018            04           4   TABLE [illegible] AL--
--------------------------------
```

```
 7019  1B6B      23          35   ONE ENTRY FOR EACH ALPHA
INC            HL
--------------------------------
 7020  1B6C      10          16   CHARACTER.  LOOP UNTIL
DJNZ           $ -5
 7021            FB         251   JUMP TO 7017 TABLE DONE
================================
 7022  1B6E      AF         175   ZERO ACCUMULATOR
XOR            A
--------------------------------
 7023  1B6F      32          50   #78F2 = 30962:
LD             (#78F2),A
 7024            F2         242   FLAG NO ERROR FOR RESUME
 7025            78         120
--------------------------------
 7026  1B72      6F         111   ZERO
LD             L,A
--------------------------------
 7027  1B73      67         103   HL REGISTERS
LD             H,A
--------------------------------
 7028  1B74      22          34   #78F0 = 30960:
LD             (#78F0),HL
 7029            F0         240   ZERO ON ERROR ADDRESS IN
 7030            78         120   #78F0/F1 OF COM. REGION
--------------------------------
 7031  1B77      22          34   #78F7 = 30967: ZERO
LD             (#78F7),HL
 7032            F7         247   POINTER TO NEXT STATEMENT
 7033            78         120   FOLLOWING BREAK,STOP,END
--------------------------------
 7034  1B7A      2A          42   #78B1 = 30897:
LD             HL,(#78B1)
 7035            B1         177   GET TOP OF MEMORY
 7036            78         120
--------------------------------
 7037  1B7D      22          34   #78D6 = 30934: LOAD
LD             (#78D6),HL
 7038            D6         214   POINTER TO NEXT AVAILABLE
 7039            78         120   PLACE IN STRING SPACE
--------------------------------
 7040  1B80      CD         205   #1D91 = 7569:
CALL           #1D91
 7041            91         145   CALL RESTORE ROUTINE TO
 7042            1D          29   PUT 31464 IN DATA POINTER
================================
 7043  1B83      2A          42   #78F9 = 30969:
LD             HL,(#78F9)
 7044            F9         249   GET END OF BASIC PROGRAM-
 7045            78         120   PREVIOUSLY SET TO 31467
--------------------------------
 7046  1B86      22          34   #78FB = 30971: SET END
LD             (#78FB),HL
 7047            FB         251   SIMPLE VARIABLES POINTER
 7048            78         120   EQUAL TO END OF PROGRAM
--------------------------------
 7049  1B89      22          34   #78FD = 30973:
LD             (#78FD),HL
 7050            FD         253   LIKEWISE WITH END OF
 7051            78         120   ARRAY VARIABLES POINTER
--------------------------------
```

IPL SEQUENCE DECODING CONTINUED . . 27/26

```
CALLED FROM #FC IN IPL SEQUENCE

______________________________
 13444 3484      CD         205   #3FA0 = 16288: TO CHECK
CALL           #3FA0
 13445           A0         160   IF CTRL KEY WAS HELD DOWN
 13446           3F          63   WHEN VZ WAS SWITCHED ON
==============================
 13447 3487      3E          62   32 IS GREEN SCREEN CODE
LD             A,#20
 13448           20          32
------------------------------
 13449 3489      32          50   #783B = 30779: LOAD GREEN
LD             (#783B),A
 13450           3B          59   SCREEN CODE INTO 30779
 13451           78         120
------------------------------
 13452 348C      32          50   ALSO LOAD '32' INTO
LD .           (#6800),A
 13453           00           0   #6800 = 26624
 13454           68         104
------------------------------
 13455 348F      3E          62
LD             A,#3C
 13456           3C          60
------------------------------
 13457 3491      32          50   #783A = 30778: LOAD
LD             (#783A),A
 13458           3A          58                 WITH '60'
 13459           78         120
------------------------------
 13460 3494      3E          62   16 IS CURSOR FLASH COUNT
LD             A,#10
 13461           10          16
------------------------------
 13462 3496      32          50   #7841 = 30785: PUT COUNT
LD             (#7841),A
 13463           41          65   IN CURSOR FLASH COUNTER
 13464           78         120   RANGE 16 TO 1
------------------------------
 13465 3499      AF         175   ZERO ACCUMULATOR
XOR            A
------------------------------
 13466 349A      32          50
LD             (#7AAF),A
 13467           AF         175   #7AAF = 31407: ZERO THIS
 13468           7A         122   LOCATION IN COM. REGION
------------------------------
 13469 349D      21          33
LD             HL,#7AB2
 13470           B2         178   #7AB2 = 31410
 13471           7A         122
------------------------------
 13472 34A0      22          34
LD             (#7AB0),HL
 13473           B0         176   #7AB0 = 31408: LOAD 31410
 13474           7A         122   INTO 31408/9
------------------------------
 13475 34A3      3E          62
LD             A,#C9
 13476           C9         201
------------------------------
```

```
 13477 34A5      C3         195
JP             #3E37
 13478           37          55
 13479           3E          62
==============================
==============================
 15927 3E37      32          50
LD             (#787D),A
 15928           7D         125   #787D = 30845: LOAD THE
 15929           78         120   INTERRUPT EXIT IN THE
------------------------------
 15930 3E3A      3E          62   COM. REGION WITH RET CODE
LD             A,#10
 15931           10          16   16 IS CODE FOR YELLOW
------------------------------
 15932 3E3C      32          50   #7846 = 30790: PUT YELLOW
LD             (#7846),A
 15933           46          70   CODE IN GRAPHIC CHARACTER
 15934           78         120   COLOR STORE
------------------------------
 15935 3E3F      C9         201   RETURN TO #FF
RET
==============================
==============================
```

```
CALLED FROM #1B50

______________________________
 7672  1DF8      AF         175   ZERO ACCUMULATOR
XOR            A
------------------------------
 7673  1DF9      32          50   #791B = 31003:
LD             (#791B),A
 7674            1B          27   SET TRACE FLAG IN COM.
 7675            79         121   REGION TO OFF -- TROFF
------------------------------
 7676  1DFC      C9         201   RETURN TO #1BF3
RET
==============================
==============================
```

IPL SEQUENCE DECODING CONTINUED . . 27/27

```
----------------------------------
7052  1B8C       CD          205   #79BB = 31163: DOS EXIT
CALL            #79BB
7053             BB          187   IN COMMUNICATIONS REGION
7054             79          121   IPL INITIALISED TO RET
==================================
7055  1B8F       C1          193   NO PREVIOUS PUSH--TRIVIAL
POP             BC
----------------------------------
7056  1B90       2A           42   #78A0 = 30880: GET LOWER
LD              HL,(#78A0)
7057             A0          160   BOUNDARY OF STRING SPACE
7058             78          120
----------------------------------
7059  1B93       2B           43   DECREMENT TWICE AND PUT
DEC             HL
----------------------------------
7060  1B94       2B           43   RESULT IN BACKSPACED
DEC             HL
----------------------------------
7061  1B95       22           34   STACK ADDRESS STORE
LD              (#78E8),HL
7062             E8          232   #78E8 = 30952
7063             78          120
----------------------------------
7064  1B98       23           35   INCREMENT
INC             HL
----------------------------------
7065  1B99       23           35   TWICE TO RESTORE ORIGINAL
INC             HL
----------------------------------
7066  1B9A       F9          249   VALUE AND PUT THIS IN
LD              SP,HL
----------------------------------
7067  1B9B       21           33   Z80 STACK POINTER
LD              HL,#78B5
7068             B5          181   REGISTER
7069             78          120
----------------------------------
7070  1B9E       22           34   #78B3 = 30899: THEN PUT
LD              (#78B3),HL
7071             B3          179   IN POINTER TO LITERAL
7072             78          120   STRING POOL: INITIALISED
----------------------------------
7073  1BA1       CD          205   #038B = 907: TO SET
CALL            #038B
7074             8B          139   OUTPUT DEVICE TO VIDEO
7075             03            3
==================================
7076  1BA4       CD          205   #2169 = 8553: ALSO SETS
CALL            #2169
7077             69          105   OUTPUT DEVICE TO VIDEO
7078             21           33   SOME REDUNDANCY HERE
==================================
7079  1BA7       AF          175   ZERO ACCUMULATOR
XOR             A
----------------------------------

----------------------------------
7080  1BA8       67          103   THEN ZERO
LD              H,A
----------------------------------
7081  1BA9       6F          111   HL REGISTERS
LD              L,A
----------------------------------
7082  1BAA       32           50   #78DC = 30940:
LD              (#78DC),A
7083             DC          220   ZERO 'FOR' STATEMENT FLAG
7084             78          120
----------------------------------
7085  1BAD       E5          229
PUSH            HL
----------------------------------
7086  1BAE       C5          197
PUSH            BC
----------------------------------
7087  1BAF       2A           42   #78DF = 30943:
LD              HL,(#78DF)
7088             DF          223   GET START ADDRESS OF
7089             78          120   PROGRAM WHEN RUN (31464)
----------------------------------
7090  1BB2       C9          201   RETURN TO #FC
RET
==================================
```

CALLED FROM #3484: SUBROUTINE TO CHECK IF <CTRL> WAS PRESSED WHEN VZ SWITCHED ON

```
______________________________
 16288 3FA0      3A        58   #68FD = 26877
LD              A,(#68FD)
 16289           FD       253   GET BYTE FROM ROW ADDRESS IN KEYBOARD MATRIX
 16290           68       104   THAT INCLUDES CTRL KEY
------------------------------
 16291 3FA3      CB       203
BIT              2 ,A
 16292           57        87   CHECK FOR PRESS OF <CTRL> WHILE VZ IS BEING SWITCHED ON
------------------------------
 16293 3FA5      3E        62   LOAD ACCUMULATOR WITH DARK SPACE CHARACTER CODE--DARK CHARACTER FOR LIGHT SCREEN
LD              A,#20
 16294           20        32   THIS IS A PEEK/POKE CODE
------------------------------
 16295 3FA7      20        32   JUMP TO 16305 IF <CTRL> IS NOT BEING PRESSED
JR              NZ,$  +8
 16296           08         8   BIT 2 OF 26877 WILL BE ZERO IF <CTRL> IS PRESSED
==============================
 16297 3FA9      F6       246   <CTRL> PRESSED, SO CHANGE CODE TO '96'
OR              #40
 16298           40        64   WHICH IS LIGHT SPACE CHARACTER CODE AND DARK SCREEN CODE
------------------------------
 16299 3FAB      32        50   #7818 = 30744: LOCATIONS 30744/5 IN COMMUNICATIONS REGION WERE INITIALLY LOADED
LD              (#7818),A
 16300           18        24   WITH ZERO BYTES AT START OF IPL SEQUENCE
 16301           78       120   CHANGE SCREEN FLAG TO DARK, SINCE <CTRL> WAS PRESSED
------------------------------
 16302 3FAE      32        50   #7819 = 30745: CHANGE SCREEN CHARACTER FLAG TO LIGHT --
LD              (#7819),A
 16303           19        25   LIGHT CHARACTERS ON A DARK SCREEN
 16304           78       120
------------------------------
 16305 3FB1      32        50   AND LOAD CURSOR CHARACTER STORE IN COMMUNICATIONS REGION WITH
LD              (#783C),A
 16306           3C        60   SPACE CHARACTER CODE
 16307           78       120   #783C = 30780
------------------------------
 16308 3FB4      C3       195
JP              #01C9
 16309           C9       201   #01C9 = 457:
 16310           01         1   JUMP TO ROUTINE TO CLEAR SCREEN, HOME CURSOR AND SELECT MODE(0)
==============================
==============================
```